

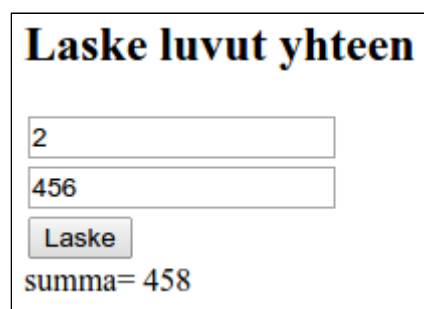
# Harjoitukset 2

- Harjoitustehtävät palautetaan [ScoreTronicilla](#). Tarjolla on myös [ScoreTronicin käyttöohjeet](#)
- Harjoitukset 2 tulee palauttaa **3.6.2020** klo 23.59 mennessä
- JavaScript-tehtävissä ei saa käyttää jQueryä tai muita vastaavia JS-kirjastoja
- Harjoituksissa 2 opit JavaScript-kielen perusteita.

## Tehtävä 1 [4p] #planone

Toteuta JavaScriptillä oheisen kuvan mukainen kahden luvun yhteenlaskin. `tulostaSumma(a, b)` -funktio ottaa parametrinaan lomakkeen kenttiin syötetyt luvut (merkkijonot) ja tulostaa lopputuloksen div-elementtiin (id=results). `tulostaSumma(a, b)` -funktio käyttää summan laskemiseen `laskeSumma(a, b)` -funktia, joka palauttaa (return) summan kokonaislukuna. Lomakkeen voit määritellä esim.

```
<form id="f">
  <input type="text" id="n1"><br>
  <input type="text" id="n2"><br>
  <button type="button" onclick="tulostaSumma(f.n1.value, f.n2.value)">Laske</b>
</form>
```



**Laske luvut yhteen**

2

456

Laske

summa= 458

### Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan

## Tehtävä 2 [4p] #planone

Toteuta JavaScriptillä oheisen kuvan mukainen taulukon alkioiden ja laskettujen suureiden tulostaja harjoitusten 1 tehtävän 6 tapaan.

## Taulukon [11, 22, 33, 44] läpikäyntiä

tulostamista ja keskeisten suureiden laskemista  
anonyymeillä funktioilla

Alkiot ovat:

```
taulukko[0] = 11  
taulukko[1] = 22  
taulukko[2] = 33  
taulukko[3] = 44
```

Lukumäärä on: 4

Summa on: 110

Keskiarvo on: 27.5

Suureiden laskennassa ja taulukon sisällön tulostamisessa tulee kirjoittaa seuraavia tavallisia **anonyymejä funktioita** käyttäen

- `arraySum` - laskee ja palauttaa alkioden summan
- `arrayAvg` - laskee ja palauttaa alkioden keskiarvon
- `arrayCount` - laskee ja palauttaa taulukon alkioden lukumäärän
- `printArray` - palauttaa taulukon sisällön visuaalisesti näytettäväksi web-sivulle neljänä riviä, joista ensimmäinen on `taulukko[0] = 11`

Kirjoita lisäksi tavallinen nimetty `writeResults()` -funktio, joka liittää tiedot taulukosta div-elementtiin (`id=results`) käyttäen apunaan em. funktioita. Alusta taulukon alkioden arvot `writeResults()` -funktiossa, jota voit kutsua ohjelmasi lopuksi `window.onload`-metodilla. Tässä tehtävässä EI saa käyttää `var`-sanalla määriteltyjä muuttujia, vaan niiden tilalla tulee käyttää ES6-syntaksin mukaisia `let`- ja `const`-määrittelyjä tilanteissa, joihin ne sopivat. Skriptiosuutesi näyttää osin esim. tältä:

```
let arraySum = function(arr) {  
    ...  
    return s;  
}  
...  
function writeResults() {  
    ...  
}
```

```
...  
window.onload = writeResults;
```

## Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan

---

## Tehtävä 3 [4p]

Toteuta JavaScriptillä edellinen tehtävä (H02T02) seuraavilla muutoksilla:

Toteuta funktiot `arraySum`, `arrayAvg`, `arrayCount` ja `printArray` lambda-merkittynä funktioina siten että ainoastaan `printArray`-funktiossa on sallittua käyttää `return`-lausetta näkyviin kirjoitettuna. Käytä `arraySum`-funktiossa [Array.prototype.reduce\(\)](#)-metodia

Muodosta lisäksi JavaScriptillä näytettäväksi generoitava sisältö Template-literaalina `sisalto`-nimiseen muuttujaan, jonka liität sitten näytettäväksi P-elementtiin. Ohessa runkoa tämän jälkimmäisen kohdan ratkaisemiseksi

```
let sisalto = `  
Alkiot ovat:<br><br>  
...  
`;  
para.innerHTML = sisalto;
```

## Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan

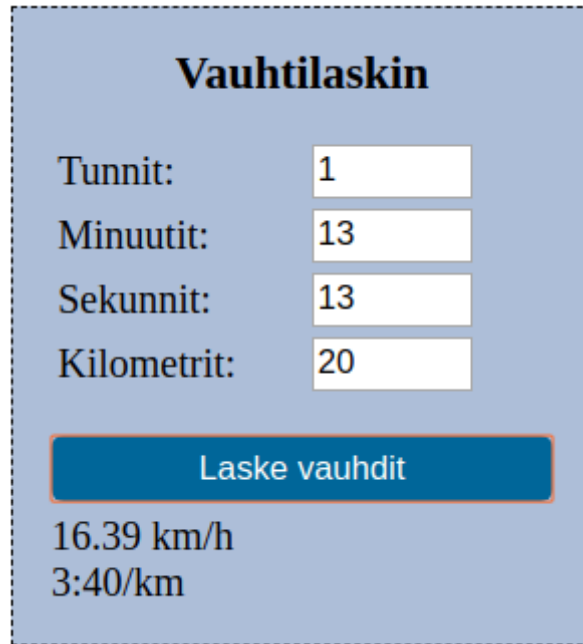
---

## Tehtävä 4 [4p]

Toteuta JavaScriptillä oheisen kuvan mukainen vauhtilaskin. Ohjelma laskee syötettyjen arvojen perusteella liikkujalle sekä km/h- että min:sec/km-vauhdin. Tulokset näytetään kahden desimaalin tarkkuudella. Lomakkeessa voi olla syötteille sopivat oletusarvot ja niille ei tarvitse tehdä tarkistuksia. Erikoisille syötteille ei tarvitse myöskään muotoilla tulostuksia, vauhti voidaan tulostaa vaikka 3456:45/km

Toteutus tulee tehdä `Vauhtilaskin`-luokkana [class-määreellä](#) määriteltynä. Syötetyt arvot tulee asettaa constructor-metodissa luokasta luotavan olion ominaisuuksiksi. Vauhdit tulee laskea metodeilla `calcKmhPace()` ja `calcMinkmPace()`. Vauhdit palautetaan "getter"-metodeilla `kmhpace()` ja

minkmpace()). Käyttöliittymästä tulee löytyä oheisessa kuvassa nähdyt osat, mutta ulkoasu on vapaa - halutessasi hyvin yksinkertainen.



**HUOM:** Yllä kuvassa ilmaisu 3:40/km tarkoittaa 3 minuuttia ja 40 sekuntia per kilometri.

Laskennassa voit googlauksen lisäksi hyödyntää suoraan seuraavia koodisnipettejä:

```
// kmh-laskentaa
var tunnit = parseInt(h)+(parseInt(min)*60+parseInt(sec))/3600;
var kmh = kilometrit/tunnit;
...

// min.sec/km-laskentaa
var sekunnit = parseInt(h)*60*60+parseInt(min)*60+parseInt(sec);
sekunnitPerKm = sekunnit/km;

var minutes = 0;

while (sekunnit >= 60){
    minutes++;
    sekunnit = sekunnit - 60;
}
sekunnit = Math.round(sekunnit);
sekunnit = String('0'+ sekunnit).slice(-2);
```

## Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan